IN THE CLAIMS

Please amend the claims as follows.

1. (Previously Presented) An apparatus comprising:

at least one processor;

a memory coupled to the at least one processor;

a first compilation unit residing in the memory, the first compilation unit comprising a plurality of object oriented classes that are part of an object oriented program, wherein the object oriented program is defined by the combination of the first compilation unit and at least one other compilation unit; and

a compiler residing in the memory and executed by the at least one processor in a partial compilation environment, the compiler allocating at least one object in the first compilation unit to an invocation stack frame for a method in the first compilation unit that allocates the at least one object, wherein the compiler comprises:

an escape analysis mechanism that operates on the first compilation unit prior to a second compilation unit and that marks each instruction in the first compilation unit that allocates a new object as one of global escape, no escape, and arg escape based on information available from classes visible in the first compilation unit but not visible in the uncompiled second compilation unit; and

an object allocation mechanism that allocates at least one object that is created by an instruction marked as no escape by the escape analysis mechanism to an invocation stack frame for a method that allocates the object.

2. (Cancelled)

3. (Previously presented) The apparatus of claim 1 wherein the escape analysis mechanism marks each instruction in the first compilation unit that allocates a new object as one of global escape, no escape, and arg escape based on information available from classes visible in the first compilation unit and from classes that are outside the first compilation unit that are visible in a specified classpath.

4. (Previously Presented) An apparatus comprising:

at least one processor;

a memory coupled to the at least one processor;

a first compilation unit residing in the memory, the first compilation unit comprising a plurality of object oriented classes that are part of an object oriented program, wherein the object oriented program is defined by the combination of the first compilation unit and at least one other compilation unit; and

a compiler residing in the memory and executed by the at least one processor in a partial compilation environment, the compiler allocating at least one object in the first compilation unit to an invocation stack frame for a method in the first compilation unit that allocates the at least one object

wherein the compiler comprises:

a code generator that creates two versions of code for a selected object method, a first version using stack allocation of objects and a second version using heap allocation of objects; and

a run time code selector that selects one of the first and second versions to execute at run time based on a determination of whether classes seen at run time match expected classes within predetermined limits.

5. (Previously Presented) An apparatus comprising:

at least one processor;

a memory coupled to the at least one processor;

a first compilation unit residing in the memory, the first compilation unit comprising a plurality of object oriented classes that are part of an object oriented program, wherein the object oriented program is defined by the combination of the first compilation unit and at least one other compilation unit; and

a compiler residing in the memory and executed by the at least one processor in a partial compilation environment, the compiler comprising:

an escape analysis mechanism that operates on the first compilation unit prior to a second compilation unit and that marks each instruction in the first compilation unit that allocates a new object as one of global escape, no escape, and arg escape based on information available from classes visible in the first compilation unit but not visible in the uncompiled second compilation unit and from classes that are outside the first compilation unit that are visible in a specified classpath;

an object allocation mechanism that allocates at least one object that is created by an instruction marked as no escape by the escape analysis mechanism to an invocation stack frame for a method that allocates the object;

a code generator that creates two versions of code for a selected object method, a first version using stack allocation of objects and a second version using heap allocation of objects; and

a run time code selector that selects one of the first and second versions to execute at run time based on a determination of whether classes seen at run time match expected classes within predetermined limits.

6. (Previously Presented) A method for allocating objects to memory in an object oriented program that comprises a first compilation unit and a second compilation unit, the method comprising the steps of:

(A) compiling the first compilation unit;

(B) during the compiling of the first compilation unit and before the compilation of the second compilation unit, marking each instruction that allocates a new object as one of global escape, no escape, and arg escape based on information available from classes in the first compilation unit and from classes that are outside the first compilation unit that are visible in a specified classpath; and

allocating at least one object that is created by an instruction marked as no escape by the escape analysis mechanism to an invocation stack frame for a method that allocates the at least one object.

7. (Cancelled)

8. (Original) The method of claim 6 wherein step (B) comprises the steps of:

creating two versions of code for a selected object method, a first version using stack allocation of objects and a second version using heap allocation of objects; and

selecting at run time one of the first and second versions to execute at run time based on a determination of whether classes seen at run time match expected classes within predetermined limits.

9. (Original) In an object oriented computer program that comprises a first compilation unit and at least one other compilation unit, a method for allocating objects in the first compilation unit to memory, the method comprising the steps of:

marking each instruction that allocates a new object as one of global escape, no escape, and arg escape based on information available from classes in the first compilation unit and from classes that are outside the first compilation unit that are visible in a specified classpath;

creating two versions of code for a selected object method, a first version using stack allocation of objects and a second version using heap allocation of objects; and

selecting at run time one of the first and second versions to execute at run time based on a determination of whether classes seen at run time match expected classes within predetermined limits.

10. (Currently amended) A program product comprising:

a compiler that compiles in a partial compilation environment a first compilation unit comprising a plurality of object oriented classes that are part of an object oriented program, wherein the object oriented program is defined by the combination of the first compilation unit and at least one other compilation unit, the compiler allocating at least one object in the first compilation unit to an invocation stack frame for a method in the first compilation unit that allocates the at least one object;

wherein the compiler comprises:

an escape analysis mechanism that operates on the first compilation unit prior to a second compilation unit and that marks each instruction in the first compilation unit that allocates a new object as one of global escape, no escape, and arg escape based on information available from classes visible in the first compilation unit; and

an object allocation mechanism that allocates at least one object that is created by an instruction marked as no escape by the escape analysis mechanism to an invocation stack frame for a method that allocates the object;

wherein the escape analysis mechanism marks each instruction in the first compilation unit that allocates a new object as one of global escape, no escape, and arg escape based on information available from classes visible in the first compilation unit but not visible in the uncompiled second compilation unit and from classes that are outside the first compilation unit that are visible in a specified classpath; and

<u>recordable</u> signal bearing media bearing the compiler.

11. (Cancelled)

12. (Cancelled)

13. (Cancelled)

14. (Cancelled)

15. (Original) The program product of claim 10 wherein the compiler comprises:

a code generator that creates two versions of code for a selected object method, a first version using stack allocation of objects and a second version using heap allocation of objects; and

a run time code selector that selects one of the first and second versions to execute at run time based on a determination of whether classes seen at run time match expected classes within predetermined limits.

16. (Currently amended) A program product comprising:

(A) a compiler that compiles a first compilation unit comprising a plurality of object oriented classes that are part of an object oriented program, wherein the object oriented program is defined by the combination of the first compilation unit and at least one other compilation unit, the compiler comprising:

(A1) an escape analysis mechanism that marks each instruction that allocates a new object as one of global escape, no escape, and arg escape based on information available from classes in the first compilation unit and from classes that are outside the first compilation unit that are visible in a specified classpath;

(A2) an object allocation mechanism that allocates at least one object that is created by an instruction marked as no escape by the escape analysis mechanism to an invocation stack frame for a method that allocates the object;

(A3) a code generator that creates two versions of code for a selected object method, a first version using stack allocation of objects and a second version using heap allocation of objects; and

(A4) a run time code selector that selects one of the first and second versions to execute at run time based on a determination of whether classes seen at run time match expected classes within predetermined limits; and

(B) recordable signal bearing media bearing the compiler.

17. (Cancelled)

18. (Cancelled)